DEC STD 095-0 VAX/VMS Internal Application Release Criteria

DOCUMENT IDENTIFIER: A-DS-EL00095-00-0000 Rev A, 26-Jun-1991

ABSTRACT: This standard defines the minimum criteria required of VAX/VMS information system applications and tools to ensure proper release, installation, and operation of application software within Digital.

APPLICABILITY: This standard is mandatory for all internal business application programs intended for use in VAX/VMS environments. The contents of this document are effective 26-Dec-1991.

STATUS: APPROVED 26-Jun-1991; use VTX SMC for current status.

This document is confidential and proprietary, and is the property of Digital Equipment Corporation. It is an unpublished work protected under the Federal copyright laws.

© Digital Equipment Corporation. 1991. All rights reserved.

DEC STD 095-0 VAX/VMS Internal Application Release Criteria

DOCUMENT IDENTIFIER: A-DS-EL00095-00-0000 Rev A, 26-Jun-1991

REVISION HISTORY: REV A, 26-Jun-1991

Document Management Category:	DIS Data Management (TSD)
Responsible Department:	Information Management & Technology
Responsible Person:	John Newman
SMC Writer:	Eric Falkof

APPROVAL: This document has been reviewed and recommended for approval by the General Review group for its category.

John Newman - Corporate Finance Information Management & Technology

---

Eric Williams - Standards Process Manager

Direct requests for further information to:

Joe Donovan	U.S. Release Management
Paul Farrel	GIA Release Management
John Harrison	European Release Management
John Newman	Corporate Finance Information Management & Technology

Use $ VTX ELF for the latest location information.

Use VTX SMC to order copies of this document from Standards and Methods Control. Send distribution questions to JOKUR::SMC or call DTN: 234-4423.

CONTENTS

1 INTRODUCTION 1
1.1 PURPOSE 1
1.2 SCOPE 1
1.3 SPONSORSHIP 1
1.4 DOCUMENT ORGANIZATION 2
2 IDENTIFICATION AND VERSION CONTROL 3
3 DISTRIBUTION 5
4 INSTALLATION 8
5 ENVIRONMENT 11
6 BACKUP AND RECOVERY 17
7 DOCUMENTATION 18
8 SECURITY 19

Appendix A REFERENCED DOCUMENTS 21
A.1 EL-Class Digital Documents 21

Appendix B RELATED DOCUMENTS 23

TABLES

1 Participation 1
2 Naming and Version Numbering 3
3 Physical Packaging and Distribution 5
4 Installation Time and Procedure 8
5 Operating Environment 11
6 Backup and Restore 17
7 Documentation Set 18
8 Security 19

1 INTRODUCTION

The contents of this document are effective 26-Dec-1991.

This document provides the information system release criteria agreed to by all participating organizations for VAX/VMS internal business application software. These criteria apply to all internally developed software and all custom developed third party software for use within Digital for data and information processing.

In addition to these criteria, adherence to other standards may be required of internal business application software. For instance, adherence to *DEC STD 065 Digital Standard Data Definitions - Policy and Requirements* during the creation of data object definitions is assumed and is not addressed within this standard. Other individual participating geographical or functional organizations may have specific acceptance criteria that would be required for an installation application in their organization. Some of these standards are listed in Appendix B.

1.1 PURPOSE

This document provides the definition of the minimum application characteristics that are required to ensure proper release, installation, and operation of internal business applications software within Digital.

1.2 SCOPE

This standard is mandatory for software developers working on production kit releases, that is, software that will be released for installation and subsequent production. Adherence to this standard is not required for software developers working on development kit releases, that is, software environments that will be transferred for additional development by another organization.

Adherence to this standard **is** required of third parties that custom develop software and all Digital developers of applications software.

Adherence to this standard is not required of third party packaged software applications.

1.3 SPONSORSHIP

This document has been accepted for use by the following Digital functional and geographical organizations.

Table 1: Participation

Organization	Approved By
Corporate Finance IS	John Newman
Corporate Information Systems	Jim Regan
Europe	John Harrison
	Dave Tough
GIA	Paul Farrell

Table 1 (Cont.): Participation

Organization	Approved By
IPSG	Joe Reppucci
U.S. Area	Joe Donovan
	Ramesh Pattni

1.4 DOCUMENT ORGANIZATION

The individual acceptance criteria (or standards) are categorized by section within this document. Within each section, the standards are listed alphabetically.

For each criterion listed, the text of the standard is printed in **boldface** type, followed optionally by examples or additional descriptive information. The three-digit number in the last column is used in the installation kit autotest tool facility developed by European Release Management.

2 IDENTIFICATION AND VERSION CONTROL

This section defines the criteria related to the naming and version numbering of applications within Digital.

Table 2: Naming and Version Numbering

Name	**Standard Item** and Description	Autotest Reference Number
Facility Code	**Applications will be identified by a Facility Code (FAC) registered with Software Quality Management (SQM)** The facility code will be used when a unique application identifier is required for directory names, logical names, or any other VMS related naming situation. The facility code will be an alphanumeric string of no less than three characters and no more than nine characters.	201
Release Types	**Three types of production releases are recognized:** 1. **Functional** 2. **Upgrade** 3. **Patch** Functional releases often involve significant functional enhancements or changes to user interfaces or user procedures. Functional releases are also known as Major, Development, or Version releases. Upgrade releases generally involve minor functional enhancements and changes in internal processing. Upgrade releases are also known as Minor, Maintenance, or Point releases. Patch releases are generally mechanisms to quickly correct specific deficiencies in already released software.	203

Table 2 (Cont.): Naming and Version Numbering

Name	**Standard Item** and Description	Autotest Reference Number
Version Number	**Executable images must be linked with the identification field set to the facility code and the version number.** The Linker /OPTIONS file mechanism provides the following attribute: identification = "FAC V3.1" The identification attribute causes the 15 character identification string, including the version number, to be stored directly in the executable image, allowing the version number to be displayed through the DCL ANALYZE/IMAGE command. Applications which include third party packaged software would not be expected to relink the included software solely to embed the application's identification string. The identification attribute contents should be of the format FAC V<VV>.<U><A>, where <VV> is the one or two-digit, left justified numeric in the range of 1-99, <U> is the upgrade number in the range of 1-9 or 0 for a functional release, and <A> is an optional single upper case alphabetic character in the range of A-Z representing the patch level, if any. For example, ABC V10.1A, where ABC is the facility code 10 is the Development (Major) release number 1 is the Upgrade number A is the Patch level This standard does not prevent applications from supporting additional mechanisms. Applications remain free to support additional mechanisms for supplying their version number, for example, through a logical name FAC$VERSION, a SHOW VERSION command, or the display of the version number on menu and data entry screens. In fact, such mechanisms may be required by geography or function specific standards.	202

3 DISTRIBUTION

This section describes the criteria that are related to the physical packaging and distribution of application software.

Table 3: Physical Packaging and Distribution

Name	**Standard Item** and Description	Autotest Reference Number
Documentation	**The installation guide, operations guide, and release notes must be in separate files.** Using separate files allows these files to be copied over the network and printed independently.	301
Functional Release Kit Naming	**A Functional Release Kit must be identified by a name formed from the Facility Code (FAC) and the version number. The version number of a Functional Release should be of the form <VV>0, where <VV> is the two-digit, zero-filled numeric in the range of 01-99.** For instance, a Functional Release Kit would be named ABC030, where: ABC is the facility code 03 is the Development (Major) release number 0 is always the upgrade number for a Functional Release	302
Kit Contents	**Only documentation, command procedures, application-specific parameter files (with preloaded default values), subdictionary files, object code, and executables are included within a kit to be released for installation on production systems. Source code may be included for only those environments where source code is interpreted directly at run-time and where no intermediate format or storage mechanism exits. For instance, DTR procedures can be supplied in a CDD/Plus DMU subdictionary file, and must, therefore, not be included as source code.** Providing linked images in the kit (without object code) saves installation time by eliminating the need for linking on the target system. This time saving is replicated each time the application is installed, and the total time saving must be balanced against the additional flexibility provided by linking on the target machine. However, VMS and its layered products are virtually always upwards compatible, so kits with linked images based on the lowest common denominator version of each required layered product can generally be used in operating environments composed of subsequent versions of VMS and the layered products. *Note: Participating geographical or functional organizations may have specific acceptance criteria that would be required for an installation application in their organization.*	303
Kit Distribution Directory	**Everything that is needed to install an application, including each saveset used, must be available in a single directory.** Therefore, there must be at least three files in this directory: 1. Installation Guide, for example, ABC010_INSTALLATION_GUIDE.TXT 2. Release Notes, for example, ABC010.RELEASE_NOTES 3. Saveset A, for example, ABC010.A 4. Operations guide	304

Table 3 (Cont.): Physical Packaging and Distribution

Name	**Standard Item** and Description	Autotest Reference Number
Kit Structure	**Saveset A must contain the installation guide, the release notes, and KITINSTAL.COM. Saveset A may also contain the software components of the application such as executable images, command procedures, and the Installation Verification Procedure (IVP). If included, incidental material, such as the project life cycle documents not required by application users, operations or system management personnel, must be in separate Savesets B-Z.** An example of the contents of Saveset A follows. • Installation guide, for example, ABC010_INSTALLATION_ GUIDE.<EXT> (where <EXT> represents TXT and at least one of PS and LN03) • Release notes, for example, ABC010.RELEASE_NOTES and ABC010_RELEASE_NOTES.<EXT> • KITINSTAL.COM • Operations guide Ordinarily, with very large kits, such as for an application made up of hundreds of executables, a good practice is to separate the application software components into more than one saveset.	305
Patch Release Kit Naming	**A Patch release kit must be identified by a name formed from the FAC and the version number. The version number of a patch release should be of the form PATCH<A>_<VV><U>, where <VV> is the two-digit, zero-filled numeric in the range of 01-99, <U> is the upgrade number in the range of 1-9, and <A> is a single upper case alphabetical character in the range of A-Z representing the Patch level.** For instance, a Patch Release kit would be named ABCPATCHB_031, where: ABC is the Facility Code 03 is the Development (Major) Release number 1 is the Upgrade number B is the Patch level	307
Source Code	**Source code is not included in a kit to be released for production use.** The source code that is compiled and linked into an application should never be included in a production kit. Sometimes, however, related source code such as sample programs (for a subroutine library) are provided. Such source level code is considered documentation, that is, it is not source code for the application or tool itself, and is not within the scope of this item.	308

Table 3 (Cont.): Physical Packaging and Distribution

Name	**Standard Item** and Description	Autotest Reference Number
Upgrade Release Kit Naming	**An Upgrade release kit must be identified by a name formed from the FAC and the version number. The version number of an Upgrade release should be of the form <VV><U>, where <VV> is the two-digit, zero-filled numeric in the range of 01-99 and <U> is the upgrade number in the range of 1-9.** For instance, an Upgrade Release kit would be named ABC031, where: ABC is the facility code 03 is the Development (Major) release number 1 is the Upgrade number	306
VMSINSTAL Guidelines	**Release kits must be built according to the rules described in the document *VMS Developer's Guide to VMSINSTAL*.** All applications must be installed using the VMSINSTAL utility. VMSINSTAL is the standard VMS utility for installation of software systems and VMS. Software systems installed using VMSINSTAL are bundled into VMS BACKUP Save Set file(s) along with a procedure called KITINSTAL.COM. KITINSTAL.COM is a DCL procedure that directs the installation of the software package through the creation of directories, verification of required product versions, verification of SYSGEN parameters, the copying of files from the SYS$UPDATE directory (where VMSINSTAL restores the Save Set file), the execution of Installation Verification Procedures, and many other tasks as required. VMSINSTAL also supplies a number of standard routines that may be used by KITINSTAL.COM to support these activities.	309

4 INSTALLATION

This section describes the criteria required of the application at installation time, and especially criteria specific to the installation procedure.

Table 4: Installation Time and Procedure

Name	**Standard Item** and Description	Autotest Reference Number
Callbacks	**Within VMSINSTAL procedures, VMSINSTAL Callbacks must be used in preference to equivalent or similar DCL commands.** For instance: The VMSINSTAL callback 'ASK' must be used in preference to the DCL 'INQUIRE' statement because only the use of the VMSINSTAL callback ensures that the response is recorded in the installation log file.	401
Clean-Up	**Upon completion, installation procedures will remove from the system temporary work files that are created and required only by the installation process.**	402
Documentation	**All questions asked by the installation procedure must be fully documented in the installation guide. The operations guide must contain full instructions about application removal and transport.**	403
Hardware Independence	**Installations of applications must be supported for all valid VAX/VMS and VAXcluster system configurations with the possible exception of limitations due to specialized peripherals.** At a minimum, the design of the application or installation kit must not preclude the use of the application on more than one node in a VAXcluster. Ideally, applications can be designed in such a way as to require the installation to be performed only once per cluster (while the application is usable from each node of the cluster.)	404
Hardware Prerequisites	**The installation guide must contain details on estimated disk space requirements for installation and execution, as well as any other specific hardware requirements.** For example, be specific about application software space, data space, and space for future expansion. Disk space estimates should be recorded in blocks. Assumptions about variables, such as application data volume requirements (in business or technical terms), length of data retention periods, and frequency and expected thoroughness of file reorganizations should be documented. In addition, applications should not assume that a magnetic tape drive is available locally. If a magnetic tape drive or special device is necessary, it should be clearly stated as a specific request, meaning that the application might not be suitable for a given site.	405
Human Intervention	**All questions to be answered during the installation procedure must be asked and validated at the beginning of the process. The user should then be informed that no further human intervention is required, and the installation should finish without further intervention.**	406

Table 4 (Cont.): Installation Time and Procedure

Name	**Standard Item** and Description	Autotest Reference Number
IVP	**The installation procedure must include an Installation Verification Procedure.** An Installation Verification Procedure (IVP) is logic within an installation procedure to provide some verification about whether the software installed correctly. IVPs vary in complexity from simple MAIL messages at the end of the installation to complete sample applications that are extracted from CMS libraries, compiled, linked, executed, and tested against known results. An IVP for a complex business application actually may not be able to exercise the software fully, but instead may perform a limited verification that key files have been created or copied. An IVP need not be exhaustive to be valuable; IVPs that detect a failed installation (gross errors) can often be as useful as those that attempt to prove a correct installation.	407
IVP Option	**The installation dialogue should include a question on whether to run the IVP.** The IVP should be left (for subsequent use) in the directory pointed to by the VMS logical SYS$TEST: (whether or not the IVP is run). Applications with extensive IVPs, made of multiple command streams, should create a FAC.DIR subdirectory in SYS$TEST to store the actual IVP command streams.	408
IVP Result	**No application should remove itself if it fails within the IVP.**	409
On-Line Help	**Help text must be provided for the questions asked.**	410
Privileges	**The installation must be capable of being performed by an individual logged in under any VMS username that has only the required privileges enabled and has been assigned the necessary process quotas and limits.** The VMSINSTAL utility does not require the use of any specific system manager account. Application kits should be created such that they do not require the username SYSTEM or the use of any other specific account.	411
Software Prerequisites	**The installation guide must contain a list of all prerequisite layered products, applications, tools, SYSGEN parameter values, and the VMS features that must be enabled (for example, rights lists.)** The minimum acceptable version number of each layered product must also be provided. Incorporating a list of prerequisite products by reference to an established and well-defined standard environment such as a specific version of EASE or CVMS is allowed.	412
Time Estimate	**The installation procedure (described in the installation guide) should inform the installer when all questions have been asked and should provide an estimate of how much longer the installation will require.** Providing an exact time estimate is extremely complex. It involves factors known to the kit developer, such as the size of the kit, but it also includes factors that are only available or even estimable at installation time, such as Processor Type and current System Load. Developers are encouraged to provide an estimate (for example, 15 minutes, 1 to 2 hours, and so on).	413

Table 4 (Cont.): Installation Time and Procedure

Name	**Standard Item** and Description	Autotest Reference Number
UIC Allocation	**The installation procedure must not assume UIC allocation. The procedure should prompt for UIC if it is to be used.**	414
VMSINSTAL Kit	**A full VMSINSTAL kit is required for all Functional and Upgrade releases of application software and tools.**	416
VMS Mail Utility	**Logic to send a VMS Mail message to the Release Management organization that supplied the installation kit must be included in the POSTINSTALL phase of the installation command file.** The address of the recipient must be expressed using MTS syntax (for example, MTS$::"PKO::Release Management") to provide a supported method of automatic retry and to avoid issues with incomplete NCP tables. VMSINSTAL supports three request codes, or phases, to the installation process. These are: 1. VMI$_INSTALL — perform the installation 2. VMI$_POSTINSTALL — perform work required after the installation is complete 3. VMI$_IVP — initiate the IVP The VMS Mail message is best sent during the POSTINSTALL phase. Placement here ensures that failure of the mail message does not unintentionally cause the failure of the entire installation. In addition, should such a network error cause the POSTINSTALL to fail, it (the POSTINSTALL processing) can easily be invoked again later by the system manager. An alternate method of notifying release management of the status is to send a message to the person (username) performing the installation. This person can forward that message electronically.	415

5 ENVIRONMENT

This section defines the release criteria related to the final operating environment used with an application. This area includes the start up and shut down of the application and the application's use of VMS resources such as CDD/Plus and logical name tables.

Table 5: Operating Environment

Name	**Standard Item** and Description	Autotest Reference Number
CDD$TOP Usage	**There will be no data in the Common Data Dictionary (CDD/Plus) DMU dictionary node CDD$TOP, except for pointers to other DMU dictionary nodes.** CDD/Plus is a system-wide resource. Access and protections within CDD/Plus must be controlled and managed by the local system manager. To provide the best environment for this management to occur, necessary data should be stored below the top level of the dictionary.	501
CDD/Plus DMU Dictionary Path Name	**All CDD/Plus dictionary objects for an application stored in DMU Dictionaries will be stored in or below a dictionary node whose path name is CDD$TOP.FAC.** To facilitate access to this data, a logical name may be defined that is equal to the full path name of the Application's dictionary node. For instance: DEFINE FAC$CDD CDD$TOP.FAC will produce a logical name that can be used in references to CDD data, such as: FAC$CDD.record definition or FAC$CDD.DATATRIEVE procedure	504
CDD/Plus DMU Subdictionary Usage	**All applications utilizing CDD/Plus DMU dictionaries will use the Subdictionary facility to store CDD/Plus data. The Subdictionary file will be distributed as part of the kit and will be named FAC$DICTIONARY:FAC.DIC.** Subdictionaries are a type of DMU dictionary node whose data is stored in a separate file. Use of subdictionaries provides applications with a degree of additional transportability.	505
DCL Usage	**DCL must not be used as a programming language.** Applications must not use DCL for iterative or repetitive processing of data where better alternatives exist. DCL should be used just as a job control language. DCL statements such as OPEN, DEFINE/USER, ASSIGN/USER, and READ, which do not modify data, are permissible.	502

Table 5 (Cont.): Operating Environment

Name	**Standard Item** and Description	Autotest Reference Number
Directory Usage	**Applications will be entirely contained within one rooted directory sturcture, that will be named after the Facility Code.** All applications will exist entirely within one rooted directory structure. That directory's top level name will be the Facility Code (for example, device_name:[FAC.]). Subdirectories should be used to organize the individual files within the application. Given an application XYZ, consider the following: DEFINE/EXEC XYZ$ROOT/TRANSLATION=(CONCEALED,TERMINAL) device:[XYZ.] This will create the Logical XYZ$ROOT, such that a reference to XYZ$ROOT:[HELP] would translate to device:[XYZ.HELP]. Note the qualifier /TRANSLATION = (CONCEALED, TERMINAL) in the above example. These are translation attributes for the logical and have the following effect: TRANSLATION=CONCEALED causes the equivalence device name to be concealed in any messages that refer to it. In this case, it causes the root directory structure to be viewed by the system as a device, and the subdirectories under it to appear as if they were directories on that device. TRANSLATION=TERMINAL causes the chain of Logical translations to be stopped at this point. Normally, the resulting equivalence name would be examined to determine if it is a logical name, and if it is, the equivalence name will be translated at the appropriate time. This would continue until no further translation is possible. The TRANSLATION=TERMINAL attribute would cause the process of continued iterative translation not to occur. The purpose for using rooted directory structures is flexibility. A system in production may have its root defined as a top-level directory. If that same system needs to be tested, the root could be redefined to point to a subdirectory (like device:[TEST.XYZ.]), and once the root and all of its subdirectories and their contents are copied to device:[TEST.XYZ...], no further adjustment needs to be made to make the system run properly. The use of a logical name search list for FAC$ROOT can provide the ability to spread an application across multiple disk devices. This can also be achieved by specifying a volume set as the device in FAC$ROOT.	503
Error Trapping	**Applications must intercept and recover correctly from all trappable error conditions and inform their users.**	506
Field Test Software	**Production applications shall not utilize field test versions of VMS, layered products, or internal applications and tools.** Application development can be performed using field test software only if the production application does not use field test versions of software.	507
File Logicals	**All major application-specific files must be referenced by unique logical names that incorporate the application prefix (FAC).** This allows repositioning of the files across disks by the IS support group to optimize performance.	508

Table 5 (Cont.): Operating Environment

Name	**Standard Item** and Description	Autotest Reference Number
File Optimization	**Applications must provide a facility to reorganize data files with the objectives of optimizing performance and space. This facility should be a standalone procedure that does not require human intervention and that can be submitted for batch processing.** This applies to all application files that are likely to become fragmented during normal operation. If the data structure used by the application requires exclusive access during backup, then this facility must be designed in such a way as to 1) run only when all application access has stopped, and 2) prevent new users from starting application sessions.	509
F$MODE Check	**All application command procedures, including startup and shutdown procedures, must include an appropriate examination of the mode in which the process is executing.** For instance: $ if "'f$mode( )'" .eqs. "BATCH" then goto BATCHCODE $ if "'f$mode( )'" .eqs. "NETWORK" then goto NETCODE $ if "'f$mode( )'" .eqs. "INTERACTIVE" then goto INTERCODE $ if "'f$mode( )'" .eqs. "OTHER" then goto OTHERCODE This helps avoid processes entering an error state of looping infinitely, waiting for on-line input, while executing in network or batch mode. It will also help to avoid creation of large unintended NETSERVER.LOG files.	510
Hardcoded Device Names	**No hardcoded device names will be allowed.** It is better practice to associate a logical name with a physical device, and reference the logical name in the application. This way, when a device fails, the fail-over process consists of changing the equivalence name for the logical name. This involves no code changes, and gives the data center more flexibility. In the case of the following: DEFINE FAC$INVOICE_PRINTER_1 TTA7: creates a logical name (FAC$INVOICE_PRINTER_1) that can be substituted for TTA7: in any place that a device name is required. If the device TTA7: fails, the data center may move its function to another device by redefining the logical name.	511
Logical Names	**The standard Facility Code (FAC) plus a dollar sign character ($) will be used to prefix all application-generated logical names.** In a production environment, it is often necessary to identify which application system defines a particular Logical Name. The FAC will be used for this purpose. For a system with the FAC of XYZ that has a subdirectory called [.LIB] a logical name might be defined by executing the following command: DEFINE XYZ$LIBRARY XYZ$ROOT:[LIB]	512

Table 5 (Cont.): Operating Environment

Name	**Standard Item** and Description	Autotest Reference Number
Logical Name Table	**Application generated logical names will be contained in a shareable Logical Name Table named FAC$LOGICAL_NAMES. The logical name table should be protected using an ACL to ensure that no unauthorized access is allowed.** Logical names are defined within tables. Within VMS, there are four tables that contain logical names normally available to a user's process. Those tables are named LNM$SYSTEM, LNM$GROUP, LNM$PROCESS, and LNM$JOB. When the system must translate a logical name, the tables will be searched in a predefined order until either a match for the logical name is found, or the tables have been completely searched. That order is usually: 1. The LNM$PROCESS table is defined for each process and is created at process creation time. Process table logical names are available only to the process. 2. The LNM$JOB table is defined for each job and is created at job creation time. Job table logical names are available to the process that created the table and all of its subprocesses. 3. The LNM$GROUP table and its contents are defined at the system level and are available to all jobs whose group number matches that of the LNM$GROUP table. 4. The LNM$SYSTEM table and its contents are defined at the system level and are available to all jobs on the system at all times. It is desirable to reserve the System table for use by VMS and VMS Layered Products. And, since there is no guarantee that every user will have the same Group code as the LNM$GROUP table, the Group table cannot be used. Therefore, the Application System logical names must be loaded into their own shareable Logical Name Table. An Application logical name table can be created by each application as part of its Startup command file at system startup time. There is a logical name (LNM$FILE_DEV) that is contained within the Parent Table LNM$SYSTEM_DEV that defines the search order for file and device name translation in RMS and other I/O services.	513
Message Utility	**Applications must use the VMS Message Utility to define application-specific message text, such as error messages.** Message literals that are embedded in application source code are difficult to maintain and expensive to translate into different languages. The VMS Message Utility provides a general purpose Formatted ASCII Output (FAO) definition facility that is well integrated with VMS System Service calls (for example, LIB$SIGNAL, SYS$GETMSG) and is used by VMS utilities and layered products (for example, %DIRECT-W-NOFILES, no files found).	514
Minimal Privileges	**User accounts must not need VMS privileges other than NETMBX and TMPMBX to run any application.** Executable images should be installed with the necessary privileges. This standard addresses end users and end user applications, not system management personnel and utilities.	515

Table 5 (Cont.): Operating Environment

Name	**Standard Item** and Description	Autotest Reference Number
Shutdown Procedure	**All actions required to terminate a running application will be contained in a command file named FAC$<COMPONENT_ >SHUTDOWN.COM where <COMPONENT_> may be "CLIENT_" OR "SERVER_". A single FAC$SHUTDOWN.COM must always be provided that shuts down all components of the application.** This standard requires that there shall be a method for the data center to terminate the application, if required. The shutdown process must logically shut down the application to ensure data integrity and cleanly force application users to exit. This termination could be due to scheduled maintenance, non-use of the application, or to permit a backup procedure. The application shutdown procedure should be placed in SYS$MANAGER and written in a way that it can be executed from the system shutdown procedure.	516
Startup Procedure	**All actions required to initiate an application at system startup time will be contained in a command file named FAC$**<COMPONENT_ >**STARTUP.COM.** This standard requires there be only one place in the system startup for data center personnel to go for procedures to add or remove an application from their system. This command file may be called from the SYS$STARTUP:SYSTARTUP_V5.COM file that is found in the SYS$MANAGER account of all VMS systems. An example of a statement used to execute the startup file for an application whose facility code is XYZ would be as follows: $ @SYS$MANAGER:XYZ$STARTUP In this command file, first, define the Application Logical Table Next and the Root_directory, using the device name that is passed to it. Definitions for all other Application System Logical Names should follow. Finally, after the system structure is defined, any images that require privileges should be installed and batch or background (server) jobs can be initiated.	517

Table 5 (Cont.): Operating Environment

<table>
<tr><th>Name</th><th>Standard Item and Description</th><th>Autotest Reference Number</th></tr>
<tr><td>Subdirectory Structure</td><td>The following standard subdirectories will be used:
<table>
<tr><th>Directory</th><th>Logical Name</th><th>Description</th></tr>
<tr><td>FAC$ROOT:[EXE]</td><td>FAC$SYSTEM</td><td>Executable images</td></tr>
<tr><td>FAC$ROOT:[COM]</td><td>FAC$COMMAND</td><td>Command procedures</td></tr>
<tr><td>FAC$ROOT:[DAT]</td><td>FAC$DATA</td><td>Data files</td></tr>
<tr><td>FAC$ROOT:[SCR]</td><td>FAC$SCRATCH</td><td>Scratch files</td></tr>
<tr><td>FAC$ROOT:[WRK]</td><td>FAC$WORK</td><td>Work area</td></tr>
<tr><td>FAC$ROOT:[HLP]</td><td>FAC$HELP</td><td>Help Library files (that is, *.HLB)</td></tr>
<tr><td>FAC$ROOT:[LIB]</td><td>FAC$LIBRARY</td><td>Shared files not covered in other directories. (for example, Object libraries, Shareable images,
Macro libraries, Form libraries)</td></tr>
<tr><td>FAC$ROOT:[MSG]</td><td>FAC$MESSAGE</td><td>Message Library files</td></tr>
<tr><td>FAC$ROOT:[RPT]</td><td>FAC$REPORT</td><td>Report file area</td></tr>
<tr><td>FAC$ROOT:[FDL]</td><td>FAC$FDL</td><td>File Definition Language files (that is, *.FDL)</td></tr>
<tr><td>FAC$ROOT:[DIC]</td><td>FAC$CDD</td><td>CDD subdictionary file (that is, FAC.DIC)</td></tr>
</table></td><td>518</td></tr>
<tr><td>System Service</td><td>System services and run-time library routines should be used wherever possible.
Applications should not rely on formatted ASCII text output from any VMS utility except in specific cases where no system service is available.</td><td>519</td></tr>
<tr><td>Virtual Memory Usage</td><td>Applications using run-time Virtual Memory Services directly must optimize virtual memory usage.
Use of Virtual Memory Services (such as LIB$FREE_VM, LIB$FREE_VM_PAGE) to return previously allocated virtual memory is required.</td><td>520</td></tr>
<tr><td>Work File Clean-Up</td><td>Any application in production must clean up temporary work files upon completion.</td><td>521</td></tr>
<tr><td>Work File Ownership</td><td>Applications must create listing, report, and work files in the user's personal area unless these have to be centrally located for a specific purpose.</td><td>522</td></tr>
</table>

6 BACKUP AND RECOVERY

This section describes the criteria required of an application with respect to the techniques used for backing up and restoring the data used within the application.

Table 6: Backup and Restore

Name	**Standard Item** and Description	Autotest Reference Number
Backup Procedure	**All applications must include an automated mechanism to backup critical data files.** The application procedure can best determine under which circumstances to backup the files to an Operations defined area (for example, disk or tape) after which Operations could proceed to further secure the data. Backup procedures should require no manual intervention and should be modifiable to the local environment. The backup procedure must be executable remotely over the network. If the output of the backup process is not VMS Backup Savesets that contain only all of the files to be restored, then an automated recovery procedure should also be provided.	601
Checkpointing	**Applications with long running (that is, greater than four hours) batch processes should be written so that after a system failure occurs, the batch processes can be restarted from the last checkpoint.** Application checkpointing should be performed after each major phase of processing. In many cases, application-specific requirements may dictate even greater checkpointing frequencies.	602
Critical Files	**Any application must contain a list of critical data files for backup purposes in the Operations Guide.**	603
Documentation	**The Operations Guide must provide clear directives on how to recover from network, application, and system failures.**	604
Integrity Checks	**The backup procedure must have automated checks to determine if backup can safely and correctly be performed** In some cases, application data bases cannot be backed up while data base activity is occurring. In these cases, to ensure the integrity of the backup files, the backup process must be prevented from continuing while users are still logged in or other read-write processing is performed.	605

7 DOCUMENTATION

This section provides criteria related to the documentation set associated with an application or tool.

Table 7: Documentation Set

Name	**Standard Item** and Description	Autotest Reference Number
Installation Guide	**An application must provide documentation that clearly explains each step of the installation process.**	701
Machine Readable Formats	**Applications must include all documentation in machine-readable format in ASCII (.TXT) and either .LN03 or .PS formats. Bookreader format files are also suggested.**	702
Operations Guide	**An application must provide documentation for use by operations personnel in both daily activities, such as backup, and in extraordinary circumstances, such as recovering from a system or data base failure.**	703
Release Notes	**An application must provide documentation to record any notable features, incompatibilities or other known deficiencies in the version of the software being released.**	704
Title Page	**The documentation front page must indicate the application name and the release number, in addition to the document title. This information must be consistent with the kit identification.**	705

8 SECURITY

This section describes criteria associated with securing the data used within the application from unauthorized access or modification.

Table 8: Security

Name	**Standard Item** and Description	Autotest Reference Number
Access Control	**Access control within an application should be ensured through Rights Identifiers, Access Control Lists, and Proxy Accounts.** Password or UIC-based mechanisms should not be used. Access Control Lists (ACL) offer a general purpose mechanism for associating users, protection attributes (such as read, write, delete) and system resources (such as a file of data base). Right Identifiers are a general mechanism for representing a user or group of users to the system. The identifier is alphanumeric and is typically a convenient term that symbolizes the nature of the group of users (for example, PAYROLL_USERS). While ACLs and Rights Identifiers are used to control access to local system or VAXcluster resources, Proxy Accounts are used to provide access to users on other nodes. Identified users on other nodes can be allowed to access a specific account without supplying the username and password.	801
DCL Access	**Users should only be able to modify an application data file while actually running the application (not through DCL directly or through utilities invoked outside of the application).** VMS does not currently support the direct granting of Rights Identifiers to executable images. While support for this is expected in a future release, the current suggested method is to put a Rights Identifier on the file(s), which is granted, by default, to no one. The nonprivileged user can run the application from his or her own account via another Rights Identifier on the application executable, which is installed with privileges. The application makes a privileged call to grant itself the file's Rights Identifier, and begins processing. Note that the application always needs to call to revoke the Rights Identifier upon (any) exit conditions. Many Fourth Generation Language application environments can support this requirement through 3GL subroutine calls. In other 4GL environments, some other technique of preventing application data base update by external sources (including the external use of the same 4GL in which the application is written) must be provided, or its absence evaluated in light of the business application being automated.	802
Hard Coded Passwords	**Passwords must not be hard coded into Network Access Control Strings.** When network access control strings cannot be written in the form MYNODE"JONES password":: passwords cannot be hard coded and must be supplied by the user each time the application is invoked.	804
Username	**The user must be able to access the application only when logged in under his or her personal Username.** Applications must not require the use of interactive group accounts in production mode. Applications should be accessible to users via appropriate ACLs to the users' personal accounts.	803

Table 8 (Cont.): Security

Name	**Standard Item** and Description	Autotest Reference Number
World Trap ACE	**A World Trap ACE (Access Control Entry) is required at the end of each ACL.** The format should be (IDENTIFIER=*, OPTIONS=PROPAGATE,ACCESS=NONE) Without this, users who have not matched an entry will revert to UIC-based protection with the possibility of unexpected results.	805

APPENDIX A REFERENCED DOCUMENTS

A.1 EL-Class Digital Documents

EL-Class Number	Document Title
EL-00065-00	*DEC STD 065-0 Digital Standard Data Definitions - Policy and Requirements*

Use VTX SMC to order copies of EL-class documents from Standards and Methods Control. Send distribution questions to JOKUR::SMC or call DTN: 234-4423.

APPENDIX B RELATED DOCUMENTS

Digital Standards

The following documents provide additional discussion of topics covered by, or related to, this document.

Corporate Security Standard No. 10.1 - Proprietary Information Protection	January, 1988
Corporate Security Standard No. 11.1 - Computer Security	30-May-1990
Software Quality Management (SQM) VMS Technical Requirements	Latest version available from SQM::SYS$INFO:SQM_VMS_ TECHNICAL_REQUIREMENTS.*
Applications Use of the DEC: Namespace - Guidelines	V1.0; latest version available from IAMOK::DNS_PROGRAM:APPL_ USE_OF_DEC_NS.PS

Geography-Specific Standards

The international use of any application must be considered and all relevant standards shall apply. However, geography-specific standards must be followed for applications used within the creating Area. Contact the applicable persons listed in Table 1 for applicable versions of these documents.

Information Services Europe Application Development Standard: Application Security Standard	V1.0	September, 1989
Information Services Europe Application Development Standard: IS Acceptance Criteria	V2.0	12 December, 1989
Information Services Europe Application Development Standard: Design Specifications Standard	V1.0t4	March, 1989
Information Services Europe Application Development Standard: Functional Specification Standard	V1.1	October, 1989
Information Services Europe Application Development Standard: Support Manual Standard	V2.0t1	November 10, 1989
Information Services Europe Application Development Standard: User Requirements	V2.0	March, 1990
GIA Release Management Standards	V0.5-3	May, 1990
USIM&T Application Environment Standards	V3-3	September 25, 1990

```
+---------------------------------------------------------------------+
|                          READER COMMENTS                            |
|                                                                     |
| Your  comments  and suggestions  will help  Standards and Methods   |
| Control improve their services and documents.                       |
+---------------------------------------------------------------------+
```

Did you request this document? _____ If so, did it arrive within a satisfactory period of time? _____ Please comment.

What are your impressions of this document? Consider format, organization, completeness, readability, and illustrations.

————————————**FOLD ON THIS LINE**———————————————————————— -

Did you find technical or clerical errors in this document? If so, please specify the page number(s) and the error(s).

Are the instructions for the update package clear? _____
Was an index available? _____ If not, is one needed? _____
Do you have other suggestions for improving this document?

The following information is optional:

Name _____ Mailstop _____

Department _____ Node _____

Send your comments to JOKUR::PROJECTS, or fold, staple, and send this page through interoffice mail to:

```
+------------------------------+
|       READERS' COMMENTS      |
| STANDARDS AND METHODS CONTROL |
|           NRO4/D4            |
+------------------------------+
```